
Table Viewer DLL™

User's Manual

Table Markup Format™ Reference

Style Markup Format™ Reference

for GUIDE® Author™



Table Viewer DLL™ User's Manual

Table Markup Format™ Reference

Style Markup Format™ Reference

All *GUIDE*® documentation and training materials are copyrighted, and all rights are reserved. Except as authorized in the terms of a valid license agreement, neither the documentation nor any software that accompanies it may be reproduced, translated, or reduced to any electronic or printed form without the prior consent of InfoAccess™ Inc.

Copyright © 1998 InfoAccess Inc. All Rights Reserved.
Printed March 1998 in the United States.

InfoAccess, the InfoAccess logo, *GUIDE Table Viewer DLL*, and *GUIDE Table Viewer Style Editor* are trademarks of InfoAccess Inc.

GUIDE is a registered trademark and *GUIDE Author*, *GUIDE Indexer*, *GUIDE Professional Publisher*, *GUIDE Reader*, *GUIDE Viewer*, *GUIDE Writer*, *GUIDE Writer Style Editor* and *LOGiX* are trademarks of Office Workstations Limited licensed to InfoAccess Inc.

Other trademarks and registered trademarks are the property of their respective owners.

Information is subject to change without notice.

InfoAccess Inc.

15821 NE 8th St
Bellevue, WA 98008-3905
USA

Technical Support

Phone 425-201-1916
Email techsupp@infoaccess.com

Corporate Headquarters

Phone 425-201-1915
Sales 800-344-9737
Fax 425-201-1922
Web www.infoaccess.com
Email info@infoaccess.co

Table Viewer DLL

GETTING STARTED

BUILDING TABLES

PLACING TABLES IN GUIDE

FORMATTING WITH GUIDE TABLE VIEWER STYLE EDITOR

INDEX

Table Markup Format Reference

ABOUT THIS REFERENCE

TABLE MARKUP FORMAT TAG DESCRIPTIONS

TABLE MARKUP FORMAT DTD

Style Markup Format Reference

ABOUT THIS REFERENCE

STYLE MARKUP FORMAT TAG DESCRIPTIONS

STYLE MARKUP FORMAT DTD

Table Viewer DLL™

***User's Manual
for GUIDE® Author™***

Contents

1 GETTING STARTED

About This Manual	10
A Note About Scripts and Syntax	10
What You Need	11
How to Proceed	12

2 BUILDING TABLES

TMF and SMF Files	13
Basic Concepts	14
Table Basics	14
Cell Basics	15
Cell Selection	15
Hotspots	16
Table Structure	16
Table Body	19
Defining Table Content	21
Paragraphs	21
Lists	22
Graphics	24
Spans	25
Table Title	26
Table Footer	27
Character References	27
The Table Window	28

3 PLACING TABLES IN GUIDE

Placing a Table in GUIDE Author	29
---------------------------------------	----

4 FORMATTING WITH GUIDE TABLE VIEWER STYLE EDITOR

Style Markup Format	32
Starting GUIDE Table Viewer Style Editor	33
The Application Interface	34
Styles and Attributes	35
The Style Editor Menus	36
File Menu	37
View Menu	38
Style Menu	39
Help Menu	39
Defining Styles	39
Resolving Style Conflicts	40
Creating a Style File	40
Creating a Style	42
Style Attributes	43
Table Attributes	44
Cell Attributes	44
Paragraph Attributes	46
Graphic Attributes	46
Character Attributes	48
Border Attributes	50
Modifying a Style	51
Editing attributes	52
Deleting a Style	52
INDEX	53

CHAPTER 1

GETTING STARTED

In *GUIDE*® *Author*™ version 5.0, tables are handled by means of Table Viewer DLL™ (**guidetv.dll**), a dynamic link library that allows you to embed tables in *GUIDE*® documents and display them in a *GUIDE* window. The functionality provided by Table Viewer DLL includes the ability to zoom and pan tables, just as you would images; to add Hotspots so that they completely cover the cells of the tables they are added to; and to run serial string find functions against the table data in order to highlight an entire cell.

As you import tables (TMF files) into *GUIDE*® *Author*™ to create your *GUIDE* documents, Table Viewer DLL interprets Table Markup Format™ (TMF™) based on a Standard Generalized Markup Language (SGML)DTD (Document Type Definition). As it interprets each TMF file, Table Viewer DLL consults an associated Style Markup Format™ (SMF™) file created in *GUIDE* Table Viewer Style Editor. It then renders the table on-screen according to the styles specified in the SMF file. That is, TMF files are used to determine table input (structure, content, and behavior), while SMF files dictate table styles. Copies of the TMF and SMF DTDs can be found in the appendixes to the two references that follow this user's manual; electronic versions of the DTDs are in the DTD subdirectory of your *GUIDE* *Author* installation directory.

While Table Viewer DLL works in the background, you work formally in *GUIDE* Table Viewer Style Editor application to style table elements. With the Style Editor, you can create new styles as well as edit, save, delete, and rename existing ones. The Style Editor interprets Style Markup Format (SMF).

For specific table markup syntax, please refer to the *Table Markup Format Reference*. For specific style markup syntax, please refer to the *Style Markup Format Reference*. For more information about styling tables, see Chapter 4, "Formatting with *GUIDE* Table Viewer Style Editor."

About This Manual

This manual assumes you are familiar with *GUIDE Author* and that you have used *LOGiiX™*, *GUIDE Author's* built-in scripting language, to author *GUIDE* electronic documents.

Table Viewer DLL User's Manual includes:

- ◆ An explanation of Table Viewer concepts
- ◆ Instructions on how to use *GUIDE* Table Viewer Style Editor to apply styles to table components
- ◆ Two special reference guides: *Table Markup Format Reference* and *Style Markup Format Reference*

This manual uses the following documentation conventions:

- ◆ Application names appear in *italics*, for example, *GUIDE Author*
- ◆ Command names, menu names, file names, and parameter names, as well as dialog names, options, and fields, appear in **boldface**
- ◆ “Readers” refers to your end users
- ◆ “*GUIDE* document” refers to a file created in *GUIDE Author* or converted by *GUIDE Writer*
- ◆ Command lines and syntax examples are indented with a special font:

`<this is a sample of a syntax entry>`

A Note About Scripts and Syntax

The formatting constraints of the printed page won't allow us to show particularly long lines of syntax exactly as they appear on-screen. Where a line of syntax has been broken, the continued line returns

to the left margin, not to the indentation level where the script began. The first level of indentation shown in these pages represents the left edge of a computer screen. Also, for greater readability, we've used upper- and lower-case characters and indents to clarify syntax structure.

New Lines

```
<table columns="5" rows="6"
  stlfile="c:\gtables\tutorial.smf">
  <tttitle>
    <para>SERVICE ADJUSTMENT PROCEDURE</para>
  </tttitle>
```

Continuation Lines

On-screen, these lines would be part of the previous line; note that they return to the left margin, not the indent level where the script started.

What You Need

Table Viewer DLL (**guidetv.dll**) is installed automatically with *GUIDE Author*. For more information about installation, please see the *Welcome to...* manual, which also discusses hardware and software requirements.

The installation utility copies various dynamic link libraries (DLLs), including **guidetv.dll**, to the *GUIDE Author* directory. Some of these DLLs enable you to run Table Viewer DLL with *GUIDE* applications; you'll need others when you distribute *GUIDE* documents that contain tables. See the appendix to the *Welcome to...* manual for a list of all the files included with *GUIDE Author*, including those you must distribute with *GUIDE* publications.

A file named **guideio.dll** controls the input and output of data. If you load a *GUIDE* application with an outdated **guideio.dll** file, a 'dynalink' error occurs if you try to open a table.

How to Proceed

Basic Table Viewer concepts are discussed in Chapter 2, "Building Tables." Chapter 3, "Placing Tables in *GUIDE*," explains how to place tables in *GUIDE Author*. Chapter 4 is devoted to *GUIDE* Table Viewer Style Editor, which you use to create the style files and styles needed to determine how tables will display in your *GUIDE* documents. Also note that the *Table Markup Format Reference* and *Style Markup Format Reference* both follow this user's manual under the same cover.

CHAPTER 2

BUILDING TABLES

With Table Viewer DLL you can create interactive electronic documents that combine all the major elements found in hard copy word-processed tables and spreadsheets.

Like tables in a word-processed file, tables created with Table Viewer DLL can include titles, columns, rows, borders, graphics, and text that wraps. They can also display footnotes. But instead of just organizing information for hard-copy presentation or storing one piece of data per cell for accounting purposes, tables in *GUIDE* offer interactive possibilities through table Hotspots. You can:

- ◆ Incorporate Hotspots linked to other documents, tables, and sound files or video clips; you can also create Hotspots that reconfigure the application
- ◆ Create spans that combine several cells
- ◆ Place a table image in existing *GUIDE* documents
- ◆ Create nonuniform cell sizes

TMF and SMF Files

To display a table in *GUIDE Author* or *GUIDE Reader*, you first create a table file in Table Markup Format (TMF) and then place the TMF file in your *GUIDE* document. A table placed in *GUIDE* becomes an image that can be panned and zoomed like any other image in *GUIDE*.

TMF is a document type definition (DTD) instance that applies Standard Generalized Markup Language (SGML) rules. So a TMF file is simply an ASCII file that you create in a text editor such as Windows Notepad, using the series of TMF tags that constitute the TMF DTD. These tags determine table structure. Table content is whatever you choose to place between the appropriate tags in the TMF file. It can

be paragraphs of text, lists, or graphics. Once you create TMF files, you import them into *GUIDE* documents by using the **Place Graphic** command in *GUIDE Author* (see Chapter 3).

The tags that comprise the TMF DTD are described in the *Table Markup Format Reference*, later in this manual. The reference includes the TMF DTD. You recognize TMF files by their **.tmf** extension.

A second DTD instance is used to create files in Style Markup Format (SMF). SMF files, which are referenced by TMF files, determine how your tables are rendered on-screen. You create SMF files in *GUIDE Table Viewer Style Editor* (see Chapter 4). The tags that comprise the SMF DTD are described in the *Style Markup Format Reference*, later in this manual. This reference includes the SMF DTD. SMF files are recognized by their **.smf** extension.

Basic Concepts

Here are some key points about tables, cells, and Hotspots to keep in mind as you create tables for your *GUIDE* documents.

Table Basics

- ◆ A diagram in a *GUIDE* document can contain only one table. No other tables or graphics, such as a bitmap, are allowed. You can add Hotspots, but only on top of table cells.
- ◆ A table consists of a two-dimensional array of cells, in rows and columns.
- ◆ Table size is determined by the content of cells or by width and height overrides applied to columns and rows.
- ◆ You can add title and footer components to a table.
- ◆ You can zoom tables or reduce their size, just like any other image in *GUIDE*.

Building Tables

- ◆ You can use the **Copy** command (from the **Edit** menu) to copy the text from a selected table cell to the Clipboard. The text will be in the default *GUIDE* style. First select the table by left-clicking anywhere in the table. Then place the cursor over the table cell you want to copy, and press **Shift** and left-click again. You can then copy the cell's contents to the Clipboard and paste them into a word-processor document, say.

Cell Basics

- ◆ Table cell size is determined either by cell content or the width and height overrides that specify sizes for the columns and rows.
- ◆ Cell content can be of three types: paragraphs, lists, or graphics.
- ◆ You can use one, any two, or all three types of cell content—or none at all.
- ◆ Cell content may be clipped if you use width or height overrides that make cells too small to fully display the paragraph, list, or graphic content.
- ◆ You can create spans by combining several cells in a single row or column. (A given cell can belong to only one span.)

Cell Selection

- ◆ You select a table in *GUIDE* by left-clicking on the table to select the table. You then press **Shift** while left-clicking to select (invert) a table cell.
- ◆ You cannot select more than one picture in a table diagram (**Shift**–left-click is used for cell selection, not extended selection.)
- ◆ Clicking on any other *GUIDE* object will deselect a table cell.

Hotspots

- ◆ The **Insert Hotspot** command (**Format** menu) is grayed if you select a table but not a specific cell.
- ◆ The **Expansion**, **Expansion Button**, and **Group** commands (from the **Make** menu) are grayed when you select a Hotspot in a table.
- ◆ You cannot drag or resize a Hotspot in a table, but you can drag or resize the table itself (the whole image). This will move or resize all Hotspots so that they are still placed over the original cells (and will still work if the table is scaled, clipped or scrolled).
- ◆ You can also move or resize a table by changing the settings in the **Element Info** dialog (**Format** menu).
- ◆ You use the **Insert Hotspot** command (**Format** menu) to insert a Hotspot exactly on top of the selected table cell.
- ◆ You can add only one Hotspot to a table cell.
- ◆ You can select a special cursor style for Hotspots, so that the cursor changes when it is placed over a Hotspot in a table. Even so, you should still use the semi-opaque setting for the Hotspot to draw the reader's attention to the Hotspot. In that way, readers won't have to move their cursor over tables to locate Hotspots.

Table Structure

To create a table (TMF file), you first collect the data you want to include in the table, and then conceptualize how the data should be presented. Most, if not all, of the data will be placed in the *body* of the table, which is the key component of the table. If you want, your table can also have *title* and *footer* components.

The body of a table consists of a series of cells in *columns* and *rows*. Each cell may include one, any two, or all three types of content: paragraphs of text, lists, or graphics. All cells in a column must be the same width and all cells in a row must be the same height. Cells will be sized naturally unless you apply width and height overrides (see page 13).

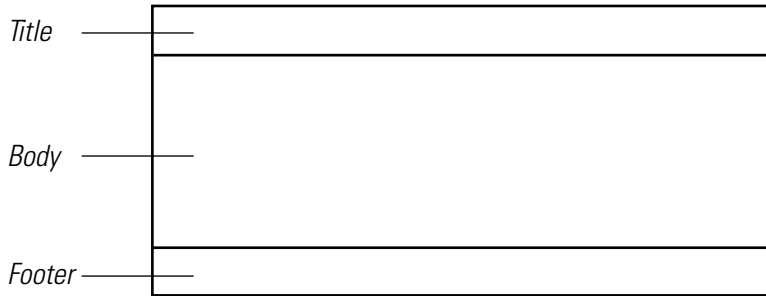


Figure 2-1
Basic table structure

To create the basic table structure shown in Figure 2-1, you would begin with the following TMF tags in the TMF file you are creating in your text editor:

```
<table>
<ttitle>
</ttitle>
<tfoot>
</tfoot>
</table>
```

The specific tags used to denote the beginning and end of a table are **<table>** and **</table>**, respectively. Every TMF file must begin and end with these tags. Optional title and footer components can be added with the **<ttitle>** and **</ttitle>** tags and the **<tfoot>** and **</tfoot>** tags.

The title, body, and footer for this table are empty because no content has been specified between the tags. For the body of a table, content can be in the form of paragraphs, lists, or graphics. You can place one, any two, or all three types of content in a given cell in the body of a table, and also in the title and footer components.

Paragraphs of texts are specified by use of the **<para>** tag; lists are specified with the **<list>** tag; and graphics are placed by use of the **<graphic>** tag. In some cases, subordinate tags are either required or are optional.

So a slightly more sophisticated table might look like:

```
<table columns="1" rows="1"
stylefile="c:\guide\samples\tables\allstyle.smf">
  <ttitle>
    <para>
      TITLE
    </para>
  </ttitle>
  <row>
    <cell>
      <para>
        BODY
      </para>
    </cell>
  </row>
  <tfoot>
    <para>
      FOOTER
    </para>
  </tfoot>
</table>
```

In this example, the TMF file specifies a simple table with one cell containing the text "BODY"; it also has a title component with the text "TITLE" and a footnote component with the text "FOOTER."

TITLE
BODY
FOOTER

Note how the two paragraph tags are essential to displaying this content in each component. The TMF file also references a style file, **Allstyle.smf** (the full path is not required if the style file is in the working directory), which provides minimal styling for the table. Finally, note the use of the **columns** and **rows** attributes immediately after the opening **<table>** tag to define the size of the table, in this case, a table of just one cell.

Table Body

The body is the heart of a table. It is composed of cells organized in columns and rows, which are specified by the **columns** and **rows** attributes in the opening **<table>** tag. The **columns** attribute defines the table horizontally (that is, sets the number of cells per row), while the **rows** attribute defines the table vertically (sets the number of cells per column). The largest possible table you can have in *GUIDE* is 40 columns by 1,000 rows.

Once you set the number of columns and rows in the table, you use the **<row>** tag to start each row you have planned for the table, and the paired **<cell>** and **</cell>** tags to define each cell in the row. When you have specified content for each of the cells in the first row (the number of cells as specified by the **columns** attribute), you use the **</row>** tag to complete the row—and are then ready to define the cells in the next row.

Note that there is no need for the cells to be numbered because Table Viewer DLL interprets the TMF file from the top down (row by row), left to right (column by column). So as long as you use the **<cell>** tag to specify the correct number of cells, as specified by the **columns** and **rows** attributes, you'll be all right.

The following table (with line spaces added to clarify syntax) demonstrates this. Here, we're using the **<para>** tag as a subordinate tag to display content within each cell. Most of the "action" in a table takes place between the **<cell>** tags, with use of the **<para>** tag, as here, or the **<list>** or **<graphic>** tag, to define content.

```
<table columns="2" rows="3"
stylefile="allstyle.smf">

  <row>
    <cell>
      <para>gearshift</para>
    </cell>
    <cell>
      <para>P42</para>
    </cell>
  </row>
```

```

<row>
<cell>
<para>clutch</para>
</cell>
<cell>
<para>P57</para>
</cell>
</row>

<row>
<cell>
<para>brake</para>
</cell>
<cell>
<para>P101</para>
</cell>
</row>

</table>

```

The above syntax would produce the following simple table in *GUIDE*:

gearshift	p42
clutch	p57
brake	p101

Figure 2-2
A simple table with two columns and three rows

One other tag worth mentioning is the **<colspec>** tag. This optional tag (which has no closing tag) is used to define columns within tables. For example, if you open the sample **Allstyles.tmf** file in a text editor (the file is in the Samples folder of your *GUIDE Author* installation directory), you will find this opening syntax:

```

<table columns="3" rows="16"
stlfile="allstyle.smf" bdrleft="Border2"
bdrright="Border2" bdrtop="Border2"
bdrbtm="Border2">
  <colspec column="1" border="Border2">
  <colspec column="2" border="Border2">
  <colspec column="3" border="Border2">

```

Here, the **border** attribute has been used to assign a specific style, `Border2`, to each of the three columns in the table. Though it isn't used here, the **width** attribute of the `<colspec>` tag is what you would use to set an override for column width. (See the *Table Markup Format Reference* for more information.)

Defining Table Content

As noted earlier, table content can consist of either regular text (defined by the `<para>` tag), lists (defined by the `<list>` tag), or graphics (defined by the `<graphic>` tag). You can place one, any two, or all three types of content in a cell. (If a cell has no content, you must still use the `<cell>` tags to define the empty cell.)

Paragraphs

A cell can contain any number of paragraphs with text aligned to the left or right of the cell or centered. You can format the font, size, color, and effects (such as bold or italic) used for text in a table cell. Text wraps to fit a cell's dimensions and can include line breaks.

- ◆ Paragraphs contain text that displays according to the font name, size, color and type (bold, etc.) defined for the style assigned to the paragraph.
- ◆ Paragraphs are left, center, or right justified according to the style assigned.
- ◆ The natural size of a cell containing a paragraph is width-to-height ratio of approximately 2:1 (a ratio found to be aesthetically pleasing).

This ratio applies only to paragraphs. However, the text of a list item is also treated as a paragraph. (This doesn't include the area occupied by the itemizer, which is minimized unless you use the **itwidth** attribute to specify the width.)

Generally, the 'natural' height of a row is the height of its tallest cell, while the 'natural' width of a column is the width of the widest cell. For example, with no width or height overrides set, the height of a particular row and width of a particular column might be set by a large graphic in a cell.

By default, the paragraph style used is the one set in MasterStyle for the SMF file that is referenced by the TMF file. But you can override the default style, if you want, by using the **style** attribute of the **<para>** tag. (The **<para>** tag is limited to this one attribute and contains no subordinate tags.)

Lists

A cell can contain a list, which you create with the **<list>** tag. You must also use the subordinate **<item>** tag to set off each entry in the list in the TMF file. The **<item>** tag, in turn, requires the use of either the **<para>**, **<list>**, or **<graphic>** tag in order to display content on-screen. (Since Table Viewer DLL lets you nest lists, the **<list>** tag can also serve as a subordinate tag to the **<item>** tag.)

The following TMF illustrates how to display a list of items:

```
<list>
<item>
<para>
fixed wheel
</para>
</item>
<item>
<para>
tilt wheel
</para>
</item>
</list>
```

Lists are defined in greater detail by several attributes: **itemizer**, **itwidth**, **separator**, and **bullechar**. **Itemizer** is the most important attribute because it specifies the type of numbered or bulleted list you want to create (see the Tag Descriptions section of the *Table Markup Format Reference*). If you don't use the **itemizer** attribute,

the list items simply display as separate paragraphs. The **itwidth** attribute works in conjunction with the **itemizer** attribute. It lets you set the width of the area the numbers or bullets occupy. The **separator** attribute specifies what comes between the number or bullet and the item itself, while the **bulletchar** attribute lets you specify the type of bullet, assuming you have selected the Bullet **itemizer**.

Any type of content (including a nested list) can be a part of a list. So the **<item>** tag uses the **<para>**, **<list>**, and **<graphic>** tags as a subordinate tags (one, two, or all three can be used). Likewise, Table Viewer resizes the column that contains that cell to accommodate the list's content.

Another extract from **Allstyle.tmf** demonstrates the syntax for a list (with indentation added to clarify syntax):

```
<cell>
  <list itemizer="Numeric" separator=".">
    <item>
      <para>
        List item 1
      </para>
    </item>
    <item>
      <para>
        List item 2
      </para>
    </item>
    <item>
      <para>
        List item 3
      </para>
    </item>
  </list>
</cell>
```

If you open **Allstyle.tmf** in *GUIDE*, you'll see that this simple list displays in the following way in the cell defined:

1. List item 1
2. List item 2
3. List item 3

Graphics

The third type of content you can have in a table is a graphic. You can place a graphic in a table's title component, in any of its body cells, or in the footer component. If you place a graphic in a single cell, Table Viewer automatically sizes the cell to accommodate the graphic—unless width or height overrides have been applied.

If the column or row that contains the graphic is limited by a width or height attribute specification, only a portion of the graphic may be visible. Since you can't scale graphics in Table Viewer, you may want to create a span of cells to accommodate large graphics.

The **<graphic>** tag can have two attributes: **style** (optional) and **file** (required). Again, the style attribute can be used to override the graphic style in the SMF file. The **file** attribute is needed to provide an external reference to the graphic file that is to be placed in the cell.

Graphics of many different formats can be placed in *GUIDE* tables. Here are some of the more popular formats:

BMP	Bitmap
JPG	JPEG
TIFF	Tagged Image File Format
PCX	PCX
PNG	Portable Network Graphics
WMF	Windows Metafile
EPS	Encapsulated Postscript
TGA	Targa
RAS	Sun Raster
WPG	WordPerfect Graphics
PCT	Macintosh Pict Format

Spans

A span is a group of adjacent cells merged into one large cell. Table Viewer DLL lets you merge cells both vertically and horizontally, but only in a single column or row. A span's content displays across all the cells of the span. Figure 2-4 displays horizontal and vertical spans.

Figure 2-3
A table with two spans

Two attributes of the `<cell>` tag are used together to create spans in a table: **sptype** and **splength**. The **sptype** attribute is set to "horiz" or "vert," while the **splength** attribute is used to designate the length of the span. For example:

```
<cell column="1" sptype="horiz" splength="6">
```

... defines a horizontal span of six cells, beginning in the first column of the row being defined. A horizontal span always runs from left to right, beginning with the cell containing the **sptype** and **splength** attributes. A vertical span runs from top to bottom, starting with the cell containing these attributes. Spans can't overlap, so a given cell can be incorporated in only one span.

Table Title

The title is an optional component that occupies the area above a table's body. Here are the essential things you need to remember about the title component of tables:

- ◆ The width of the title component is the width of the table itself.
- ◆ The height of the title component can be set in the TMF file (using the **height** attribute). The default value is the height of the content (without clipping).
- ◆ The title can contain optional and repeatable elements, where an element can be an occurrence of a paragraph, list, or graphic. (That is, you can have any number of paragraphs, lists, and/or graphics in a title component, just as you can in a cell of the table body.)
- ◆ The background color is specified by the color attribute of a cell style applied through the **style** attribute of the **<ttitle>** tag.
- ◆ The borders of a title can be set by the **bdrflag** attribute of the **<ttitle>** tag. The default value is **y1** (Yes).
- ◆ The top, left, and right borders of the title component are the same as the top, left, and right borders of the table body.
- ◆ The title can be selected in *GUIDE* by pressing **Shift** and then left-clicking on the title area.
- ◆ You can add a Hotspot to a title just as you do to any cell in the body of a table.
- ◆ You can use the **Copy** command to copy text from the title to the Clipboard once you select the title.

To place content in the title component you must use the appropriate tag, depending what kind of content you want to place. The following simple example would display the title "TITLE" in a title component with no border:

```
<table columns="2" rows="1" Allstyle.smf">
<ttitle height="720" bdrflag="n1">
<para>
TITLE
</para>
</ttitle>
<row>
...
```

Table Footer

The footer is also an optional component that functions in much the same way as the title:

- ◆ The width of the footer is the width of the table itself.
- ◆ The height of the footer can be set in the TMF file. The default value is the height of the content (without clipping).
- ◆ The footer can contain optional and repeatable elements, where an element can be an occurrence of a paragraph, list, or graphic. (That is, you can have any number of paragraphs, lists, and/or graphics in a footer component, just as you can in a cell of the table body.)
- ◆ The background color is specified by the color attribute of a cell style applied through the **style** attribute of the **<tfoot>** tag.
- ◆ The borders of a footer can be set by the **bdrflag** attribute of the **<tfoot>** tag. The default value is **y1** (Yes).
- ◆ The bottom, left, and right borders of the footer component are the same as the bottom, left, and right borders of the table body.
- ◆ The footer can be selected in *GUIDE* by pressing **Shift** and then left-clicking on the footer area.
- ◆ You can add a Hotspot to a footer just as you do to any cell in the body of a table.
- ◆ You can use the **Copy** command to copy text from the footer to the Clipboard once you select the footer.

Character References

The TMF parser supports character references. This feature allows you to use any character in your document content, regardless of its implications to the parser. For example, the SGML parser's default syntax interprets a left angle bracket (<) as the signal to open a tag. So if your document's text includes left angle brackets, you must

code them to ensure the brackets are handled properly during conversion. Since HML supports SGML character references, you can code such characters easily.

In this case, replace the bracket (<) with an ampersand (&), a pound sign (#), and the ASCII or ANSI code for that character (60 for ASCII). The parser interprets **&#60** as the left angle bracket character you want to display, not as the start of a new tag. This allows you to use any character in a document without worrying about what the parser will do. You may need to use a semicolon as a delimiter to avoid confusion. For example, if you apply the usual markup to the mathematical relationship ' $2 < 3$ ', the ambiguous ' $2 \&\#603$ ' results. If you use the ' $2 \&\#60;3$ ', however, the semicolon delimits the value '60' from the value '3.'

The Table Window

Table Viewer bases the height of each row on the highest cell in the row and the width of each column on the widest cell in the column. Row and column dimensions override cell dimensions. If a row has a height attribute, it determines the height of the row; likewise, if a column has a width attribute, that attribute determines the column's width. In short, a table window's size is based on the table's total height and width. Table height is the sum of the heights of the title, body, and footer; table width is the widest of those three components.

CHAPTER 3

PLACING TABLES IN GUIDE

You use the **Place Graphic** command in *GUIDE Author*.

Placing a Table in *GUIDE Author*

- 1 **Open the *GUIDE* document in *GUIDE Author*.**
- 2 **Position your cursor in the document at the location where you want to place the table image.**
- 3 **Choose *Place Graphic* from the *Edit* menu to open the *Place* dialog.**
- 4 **Type **.tmf* in the *File Name* field or select *TMF (*.tmf)* from the drop-down *Files of Type* box.**

This limits the display to files with the **.tmf** extension. If you know the TMF file's full path and name, enter it in the **File Name** field and go to step 7.
- 5 **Use the *Look In* list to locate and open the directory that contains the TMF file you want to embed in the document.**
- 6 **Select the TMF file.**
- 7 **Click *Open*.**

Table Viewer renders the table as a graphic in *GUIDE Author* or *GUIDE Reader*. The table image is now embedded in the *GUIDE* document and can be handled just like any other graphic (it can be resized and moved, but not edited).

Occasionally, you may encounter an error message when placing a table (TMF file) in a *GUIDE* document. You can obtain information on the error very simply. As you select the file in the **Place** dialog and click **Open**, hold down the **Shift** key. A dialog appears with an option that lets you view a message explaining the error. This message will describe the first problem encountered by Table Viewer DLL in trying to open the TMF file. If there is more than one error in the TMF file, you will have to view successive messages to resolve the error.)

CHAPTER 4

FORMATTING WITH GUIDE TABLE VIEWER STYLE EDITOR

This chapter explains how to use *GUIDE* Table Viewer Style Editor to define styles for tables in your *GUIDE* publications. With the Style Editor, you create the style files—that is, SMF files—that are referenced by TMF files when you import the latter into *GUIDE* documents. Just as TMF files dictate the structure and content of the tables you display in your *GUIDE* documents, so SMF files determine how those table will appear on-screen.

With the Style Editor, you can create, edit, rename, and delete styles for your tables, and place those styles in style files. You can create any number of style files and store any number of styles in each file. The styles you specify in the Style Editor for a table are applied to the table when the table is rendered in a *GUIDE* window by means of Table Viewer DLL.

In using the Style Editor, note:

- ◆ Full keyboard control is possible by using the tab and arrow keys. Pressing **Ctrl+M** returns the focus to the title bar unless a tab for a tab dialog has focus.
- ◆ There are six style types: table, cell, paragraph, graphic, character, and border. (A “table style” is actually a combination of the five other styles and is used to define the **MasterStyle** for each style file.)
- ◆ You can assign style tags to cells, borders, paragraphs, and graphics in the TMF file.
- ◆ *GUIDE* Table Viewer Style Editor uses device independent units (1/1440 inch, or TWIPS) for all dimensions except borders, which are in pixels.

Style Markup Format

Styles are defined in SMF files. When you save styles to a style file, *GUIDE* Table Viewer Style Editor automatically applies Style Markup Format (SMF), which is the only format that Table Viewer DLL recognizes. SMF is an SGML DTD (document type definition) that Table Viewer applies to the table. The SMF DTD is included as an appendix to the *Style Markup Format (SMF) Reference* that follows this manual. An electronic version can be found in the DTD folder in your *GUIDE Author* installation directory. You can recognize SMF files by their **.smf** extensions.

Figure 4-1 shows an extract from an SMF file. Note use of the **<stlcol>** tag. Just as every TMF file opens and closes with the **<table>** tag, every SMF file opens and closes with the **<stlcol>** tag.

```
<stlcol>
...
  <charstl id='DefaultChar' fontname='Arial'
size='10' bold='n1' italic='n2' underl='n3'
strike='n4' super='n5' sub='n6'><color red='0'
green='0' blue='0'>
  </charstl>
  <charstl id='CharA' fontname='Courier New'
size='10' bold='n1' italic='n2' underl='y3'
strike='n4' super='n5' sub='n6' ><color red='0'
green='0' blue='255'>
  </charstl>
  <charstl id='CharB' fontname='Arial Black'
size='12' bold='n1' italic='y2' underl='n3'
strike='n4' super='n5' sub='n6' ><color red='255'
green='0' blue='0'>
  </charstl>
  <charstl id='CharC' fontname='Arial' size='10'
bold='y1' italic='n2' underl='n3' strike='n4'
super='n5' sub='n6'><color red='0' green='64'
blue= 128'>
  </charstl>
...
</stlcol>
```

Figure 4-1
Part of an SMF file, showing the attributes for the four character styles in the style file: *DefaultChar*, *CharA*, *CharB*, and *CharC*.

When you place a table in a *GUIDE* document, Table Viewer DLL loads the TMF file, which, in turn, references the SMF file to determine the appropriate styles to be used to render the table.

Each style you create with the Style Editor must be referenced properly in a TMF file so the TMF can successfully call that style from the SMF file. To call styles, the TMF file first must contain a path to the style file. The correct syntax is:

```
<table stlfile="file name path">
```

The **stlfile** attribute defines the full path to the style file, for example:

```
<table stlfile="k:\tables\styles\tables.smf">
```

Here are examples of how styles in the SMF file might be referenced from the TMF file:

```
<para style="left align Bold 12">
<graph style="graph left">
<cell style= "Yellow">
<row border="border1 style">
<colspec border="border1 style">
```

Starting *GUIDE* Table Viewer Style Editor

GUIDE Table Viewer Style Editor installs automatically with *GUIDE Author*. To start the Style Editor, start Windows and do one of the following:

- ◆ Click the **Start** menu and choose **GUIDE Table Style Editor** from the *GUIDE* program group.
- ◆ Click the **Start** menu, or the **File** menu in either the Windows Program Manager or File Manager, and then choose **Run**. In the **Open** text box enter the path to **gtvstyle.exe**, the Style Editor executable, and click **OK**.
- ◆ Double-click **gtvstyle.exe** in the File Manager or Windows Explorer.

The Application Interface

The *GUIDE* Table Viewer Style Editor interface consists of an application window that contains a **Styles** list box, a series of tab dialogs, and a preview window (see Figure 4-2). There is also a menu bar containing four menus, each with a series of commands.

When you open a style file, the **Styles** list shows all the styles in the style file according to which style type is currently selected. For example, with the **Cell** tab dialog open, the **Styles** list displays all cell styles in the style file. If you switch to the **Borders** tab dialog, the **Styles** list then displays all border styles in the style file. The preview window shows your changes as you assign attributes to each style.

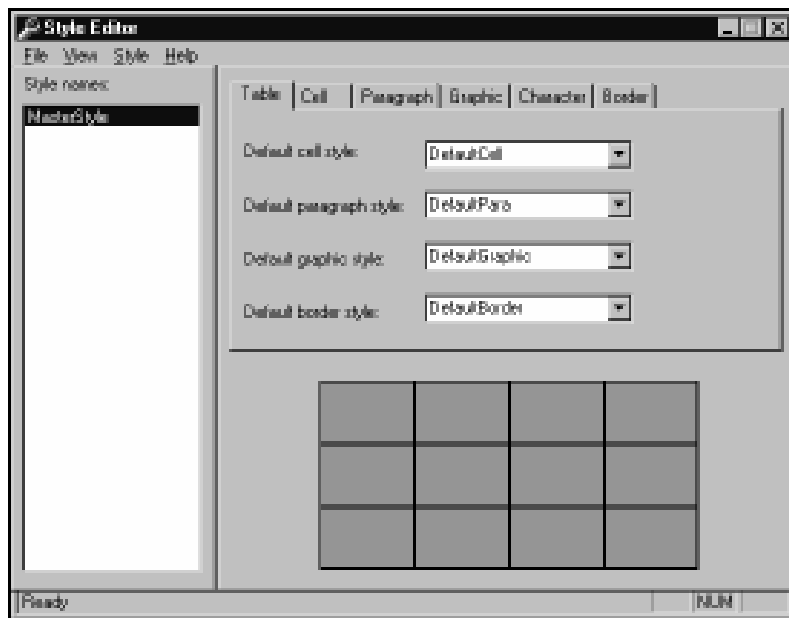


Figure 4-2
GUIDE Table Viewer Style Editor, which opens by default to the Table tab dialog

Styles and Attributes

Each new style file you create includes **MasterStyle**, which is the only style that ever appears in the **Styles** list when the **Table** tab dialog is open. **MasterStyle** is defined by the styles you choose for the various table attributes that you set in the **Cell**, **Paragraph**, **Graphic**, **Character**, and **Border** tab dialogs. You can add any number of cell, paragraph, and other styles to a style file. But you can only associate one style of each type with **MasterStyle** for a given style file. You do this in the **Table** tab dialog.

When you create a new style file, **MasterStyle** consists initially of a series of default styles (DefaultCell, etc.), which you can view in the **Table** tab dialog. (The character style is treated as an “attribute” of a paragraph style and becomes part of **MasterStyle** through its association with a specific paragraph style.) By creating a series of style files, each with its own unique **MasterStyle**, you can simplify references to styles from TMF files and ensure consistency in how tables are displayed in your *GUIDE* documents. At the same time, you can still apply unique attributes to any cell in a table by the use of specific references in the TMF file.

In other words, you may initially want to create a style file containing every style you are ever likely to need for your *GUIDE* publications. You could then decide how many distinct table styles—that is, how many **MasterStyles**—you will need to render your tables. If, say, you need only three distinct table styles, you could make copies of the original style file and customize those style files by editing **MasterStyle** in each case. That is, you could choose different cell, paragraph, and other styles from the complete stable of styles in the style file to create a unique **MasterStyle** for each style file.

Following are the types of styles you can create for each style file and the attributes or options you can select in the appropriate tab dialog in each case:

<i>STYLE</i>	<i>ATTRIBUTES</i>
Table	Not a style per se. Used to select existing styles for the treatment of cells, paragraphs, graphics and borders in the table. What you select defines MasterStyle for the style file.
Cell	Select the background color.
Paragraph	Select a character style and set text alignment and indentation.
Graphic	Set the alignment of graphics within table cells.
Character	Select a font name, point size, font style (bold, italic, underline, strikethrough, superscript, subscript), and also the foreground color (for text within cells).
Border	Select the width and color of table borders.

The *GUIDE* Table Viewer Style Editor interface reflects the above grouping of table elements and attributes. With this brief introduction to Style Markup Format and styles, let's start *GUIDE* Table Viewer Style Editor and review the application interface in more detail before we discuss the creation of styles and style files.

Note: You can move between tab dialogs as you set attributes for each table element. Choose **Save** from the **File** menu when you are ready to save your changes to the style file you're working in.

The Style Editor Menus

The Style Editor menus contain commands that enable you to create and manage style files and their styles. A menu command followed by ellipsis points (...) indicates that a dialog with further options opens when you choose that command.

To open a menu or activate a command, click on its name. Alternatively, you can use keyboard shortcuts to activate menus and commands by pressing the **Alt** key and access keys in sequence. Access keys are the underlined letters in menu and command names. For example, **Alt** followed by **F** opens the **File** menu. When the **File** menu opens, you can choose **S** to activate the **Save** command. In this documentation, keyboard shortcuts appear with commas between the **Alt** and access keys. Thus, **Alt,F,S** represents the complete keyboard shortcut for saving a style file. Alternatively, you can use the shortcuts stated opposite many commands in the pull-down menus. For example, **Ctrl+S** will also activate the **Save** command if you first press and hold down the **Ctrl** key and then press **S**.

File Menu

The **File** menu lists commands to help you create and manage style files:

COMMAND	SHORTCUT	DESCRIPTION
New	Alt,F,N	Opens a new style file with MasterStyle and its default settings.
Open	Alt,F,O	Allows you to open an existing style file.
Save	Alt,F,S	Updates the active style file under its existing name.
Save As	Alt,F,A	Allows you to save an active style file under a new name.
Exit	Alt,F,X	Closes <i>GUIDE</i> Table Viewer Style Editor.

View Menu

The **View** menu lists commands that duplicate the tab dialogs in the application window. For example, to open the **Cell** tab dialog, you can either choose **Cell styles** from the **View** menu or click the **Cell** tab in the application window.

COMMAND	SHORTCUT	DESCRIPTION
Table styles	Alt,V,T	Opens the Table tab dialog where you can select styles for the cell, paragraph, graphic, and border elements of the table.
Cell styles	Alt,V,E	Opens the Cell tab dialog where you can specify the background color of cells.
Paragraph styles	Alt,V,P	Opens the Paragraph tab dialog where you can choose a character style and set paragraph alignment and indentation.
Graphic styles	Alt,V,G	Opens the Graphic tab dialog where you can determine the positioning of graphics within cells.
Character styles	Alt,V,C	Opens the Character tab dialog where you can choose a font name, point size, a font color, and font attributes.
Border styles	Alt,V,B	Opens the Border tab dialog where you can set color and width for cell borders.
List of styles	Alt,V,L	Returns the focus to the Styles list.

Style Menu

The **Style** menu lists commands that let you create and edit styles:

COMMAND	SHORTCUT	DESCRIPTION
Add...	Alt,S,A	Opens the Add Style dialog box, which allows you to add a new style to the active style file.
Delete	Alt,S,D	Deletes the style selected in the Styles list.

Help Menu

The **Help** menu lists the single command **About**.

COMMAND	SHORTCUT	DESCRIPTION
About	Alt,H,A	Displays a dialog that contains copyright information and the version number for <i>GUIDE</i> Table Viewer Style Editor.

Defining Styles

With *GUIDE* Table Viewer Style Editor, you can define all the styles you need to display tables in your *GUIDE* documents. The key is deciding

- ◆ How many total styles you will need for your *GUIDE* publications (cell styles, paragraph styles, etc.)
- ◆ How many style files—and therefore **MasterStyles**—you will need for the different kinds of tables you plan to display in those publications

Creating a series of style files, each with its own unique **MasterStyle**, can simplify the references you make from your TMF files and therefore the management of tabular data in your publications. At the same time, you always retain a measure of flexibility as to the treatment of tables on-screen because you always have the option of applying attributes to one or more cells in a table that are different from any attributes specified by **MasterStyle** in the style file.

Resolving Style Conflicts

Sometimes Table Viewer DLL encounters style inconsistencies between the SMF and TMF files or within the TMF file itself. This can happen if the SMF file lacks a style definition referenced in the TMF file. In that event, Table Viewer applies the **MasterStyle** (which includes default styles such as **DefaultCell**). This may also occur if a table style displays incorrectly because Table Viewer DLL reads a TMF file that references the wrong style or the wrong SMF file. Such conflicts can arise when you fail to assign unique names for table styles, TMF files, and/or SMF files. It is best to use unique names for styles and style files at all times.

Creating a Style File

To create a style file, open a new file in *GUIDE* Table Viewer Style Editor and add styles to the style file one at a time. The **Styles** box in the application window lists the existing styles in the active style file for the style type currently selected. When you open a new style file, **MasterStyle** is the only style listed. **MasterStyle** is, in effect, the default table style for a new style file. It includes other default styles such as **DefaultCell**.

To create a new style file:

1 Choose the *New* command from the *File* menu.

A new file opens with **MasterStyle** listed in the **Styles** list. The application title bar still reads **Style Editor**. If an existing style file is already open when you choose **New**, the Style Editor prompts you to save any changes to that file, closes it, and then opens a new style file.

2 Choose *Save As* from the *File* menu.

When the **Save As** dialog opens, type a unique name for the new style file in the **File Name** text box, and use the **Save In** drop-down list to identify the subdirectory where you want to store the style file. You can accept the **.smf** extension for the file name or use a different extension.

3 Choose *Save* to save the style file and return to the Style Editor application window.

The title bar now displays the name that you just assigned to the new style file.

It is best to name and save a new style file as soon as you open it. You also need to ensure that any TMF file that references the style file contains the full path and the correct style file name. (Once you name a style file, its name displays in the Style Editor title bar when the style file is open.) A typical reference in a TMF file might be:

```
<table stlfile="c:\guide\styles\my.smf">
```

You can, of course, edit any style file at any time in the Style Editor. You can add, change, or delete any of the individual styles for cell, paragraph, etc., and you can change the individual styles associated with **MasterStyle** in the style file.

Creating a Style

You can create a style at any time by opening the appropriate style file and adding the individual style in the **Cell**, **Paragraph**, or another tab dialog. You can organize the styles you create within table styles. The number of table and other styles you reference in a style file will depend on the different kinds of tables in your *GUIDE* publication.

To add a new style to a style file:

- 1 Open the style (.smf) file you want to add a new style to.**
- 2 Choose *Add* from the *Style* menu to open the Add Style dialog.**
- 3 Enter the name of the new style in the *Add Style* dialog (see Figure 4-3).**
- 4 Select one of the radio buttons to specify which table element you are creating the new style for: Cell, Paragraph, Graphic, Character, or Border.**
- 5 Click *OK* to accept the style name and return to the Style Editor application window.**

The tab dialog for the style type selected in the **Add Style** dialog immediately opens and the name of the new style is highlighted in the the **Styles** list box. The Style Editor automatically assigns default attributes to the new style. For example, if you create a new cell style, its initial attributes are those of **DefaultCell**.

- 6 With the name of the new style highlighted in the *Styles* list and the appropriate tab dialog open, you can change the style attributes from the default settings, if you want.**
- 7 Choose *Save* from the Style Editor *File* menu to save the new style to the style file.**



Figure 4-3
The Add Style dialog

Style Attributes

The attributes of each style are defined on the tab dialogs for the different table elements. Just click on a tab to open that tab dialog. You can review and change individual attribute settings on each tab dialog. To change a style's attributes, select from the options available on each tab dialog (check boxes, radio buttons, etc.). Always be sure to select the correct style from the **Styles** list before you use any of the options on the tab dialogs to change style attributes. To change the tab dialog you are working on, click on another tab in the Style Editor window.

Table Attributes

On the the **Table** tab dialog (Figure 4-4) you define the **MasterStyle** for the style file. Table “attributes” are not strictly attributes at all but, rather, the specific set of styles you want to assign to the **MasterStyle** from all the styles available in the style file. In this way you can organize the various styles you want to apply to the tables you display in your *GUIDE* documents so that those tables are rendered consistently.

Remember, when you select a paragraph style, you also select the character style associated with that paragraph style. In that way, the settings on the **Table** tab dialog capture all the styles necessary to create the **MasterStyle** for the style file.

ATTRIBUTE	DESCRIPTION
Default cell style	Select the specific cell style you want to add to MasterStyle .
Default paragraph style	Select the specific paragraph style you want to add to MasterStyle .
Default graphic style	Select the specific graphic style you want to add to MasterStyle .
Default border style	Select the specific border style you want to add to MasterStyle .

Cell Attributes

On the **Cell** tab dialog (Figure 4-5), you can define the following attributes for table cells:

ATTRIBUTE	DESCRIPTION
Background color	Specifies the background color of each table cell. You can either enter the RGB values directly in the Red, Green, and Blue fields, or click Custom Color to open the Color dialog box and select a color there.

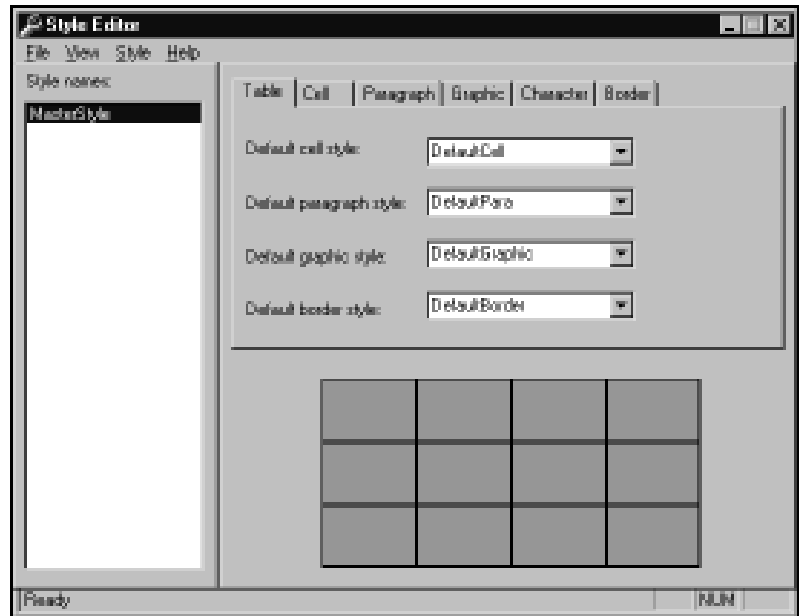


Figure 4-4 Table tab dialog

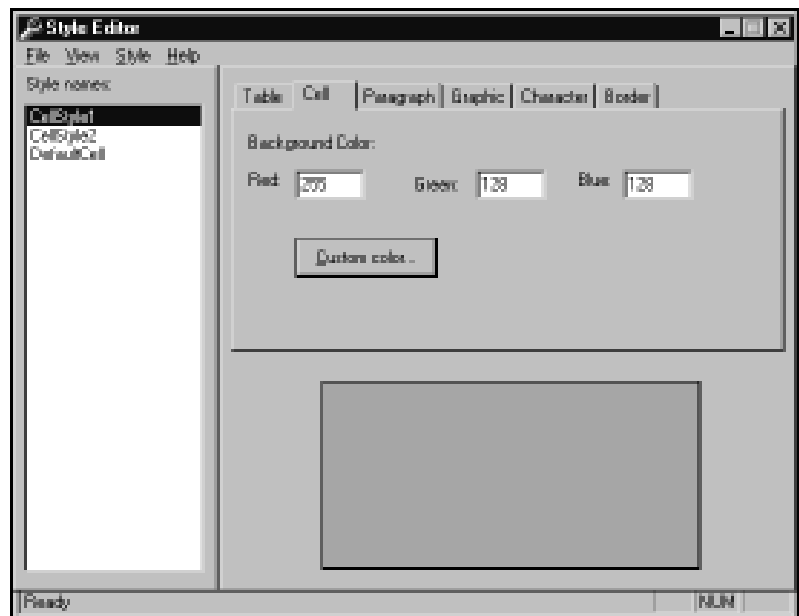


Figure 4-5 Cell tab dialog

Paragraph Attributes

On the **Paragraph** tab dialog (Figure 4-6), you can set the following attributes to define how the paragraphs will be treated:

ATTRIBUTE	DESCRIPTION
Character style	Lets you choose an existing character style to associate with the Paragraph style you set in this tab dialog. All the character styles you create in the Character tab dialog are available from this drop-down list.
Alignment	Aligns text in the paragraph to the left, right, or center.
Indents	First Line sets the indent for the first line of text in a paragraph, in TWIPs (1/1440 of an inch). Left Indent indents the whole paragraph, that is, creates a left-hand margin for text within the cell.

Graphic Attributes

On the **Graphic** tab dialog (Figure 4-7), you can define attributes for graphics that appear in table cells.

ATTRIBUTE	DESCRIPTION
Alignment	Determines the alignment of a graphic within the table cells. By default, a graphic will fill the cell completely unless extra width is added to the cell in the Extra Width text box. In that event, the alignment option you select becomes important.
Extra Width	Provides extra space between the graphic and the cell border, which can be added to one or other side of the cell (if you select Left or Right under Alignment) or to both sides (if you select Center). The dimensions are in TWIPs (1/1440 of an inch).

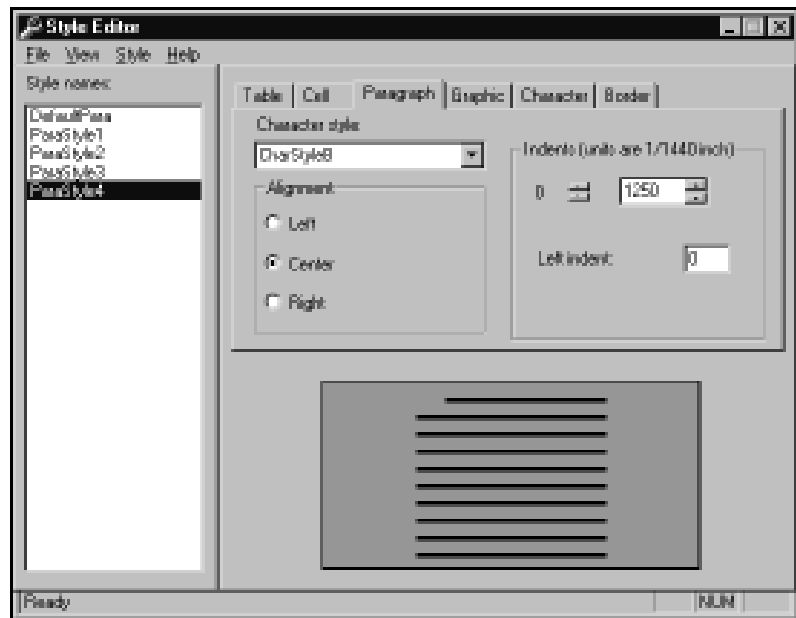


Figure 4-6 Paragraph tab dialog

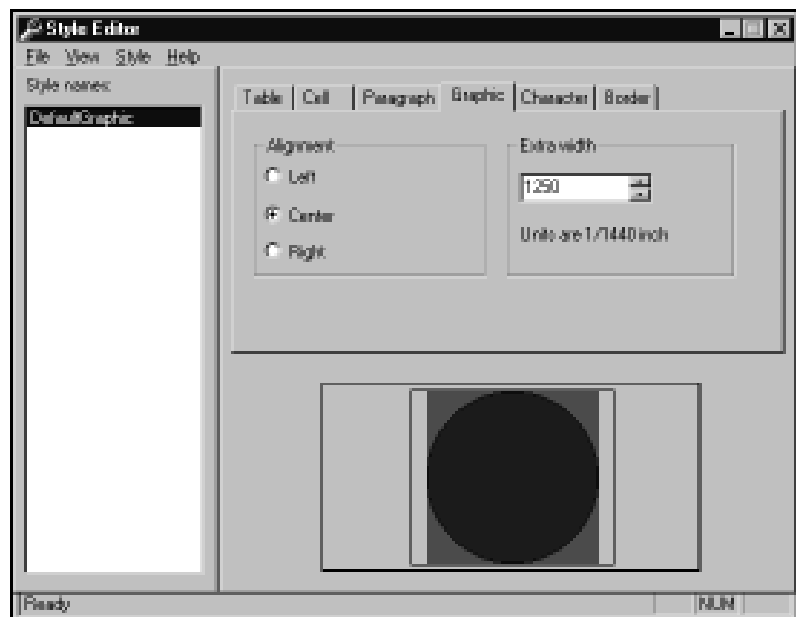


Figure 4-7 Graphic tab dialog

Character Attributes

You define the text attributes listed below in the **Character** tab dialog (Figure 4-8). The character styles that you create here only take effect for a particular table once you associate a specific character style with a specific paragraph style in the **Paragraph** tab dialog. (If you want, you can reference a specific character style in an SMF file.)

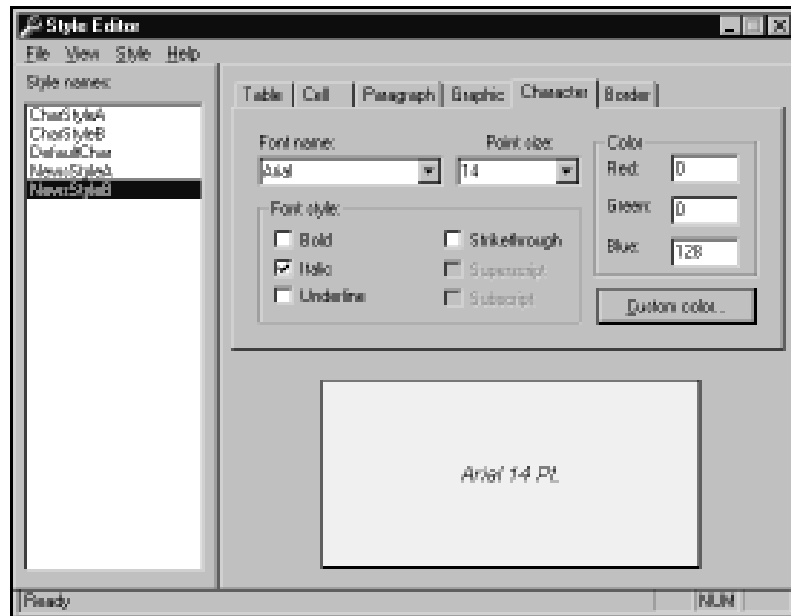


Figure 4-8
Character tab dialog

Note: The text displays about two pixels smaller when you display the table in a *GUIDE* document than the point size you select in the Style Editor. The difference will be more noticeable with smaller point sizes, and you may need to compensate for this when you choose the font size for a particular style.

ATTRIBUTE

DESCRIPTION

Font name

Sets the typeface, such as Arial, Courier, Helvetica, or Times Roman.

ATTRIBUTE	DESCRIPTION
Point Size	Sets the point size for text associated with the selected style.
Font style	Sets special effects such as bold and italic.
Color	Sets the foreground color, that is, the color of the text in your table. You can enter the RGB values in the Red, Green, and Blue fields, or click Custom Color to open the Color dialog and choose a color there.

When you select a font for a new style or change a font for an existing style, you can also choose a point size for the font. The point sizes available will be determined by the font itself and your setup in Windows. You can choose a non-system font, assuming the font is available on the computer that will display the *GUIDE* table in question.

To select a font:

- 1 Open the appropriate style file.**
- 2 Open the *Character* tab dialog, and from the Styles list, select the name of the style you want to edit.**
- 3 Select the font you want to use from the Font Name drop-down list.**

You can also type the name of the font in the text box. If you want to use a font that is not on your system, you will have to type the name of the non-system font in the text box. The Style Editor will accept the non-system font entered in the text box, but the preview box will display sample text in a default font. Non-system fonts only display properly if the proper font name has been entered in the Font Name text box and the font is installed on the computer that displays the table in your *GUIDE* publication.

- 4 Select a point size from the drop-down Font Size list.**

To select a point size that isn't in the drop-down list, you can highlight the point size already in the text box and type in the size you want. Actual sizes available for each font will be determined by the type of font installed on your machine and your Windows setup.

To create a custom color to use for text:

- 1 Click on *Custom Color* to open the *Color* dialog.**
- 2 Select a color you want to customize from the palette in the *Color* dialog.**
- 3 Click on the *Define Custom Colors* button.**

The **Color** dialog expands to show a panel that contains a full spectrum color field, a sample box that displays the color's screen display complete with dithering on the left and its solid display (the closest basic color) on the right, and text boxes that show the selected color's values for hue, saturation, luminance, red, green, and blue. The current color display and values identify the color you selected in Step 2.

- 4 Change the numbers in the *Red*, *Green*, and *Blue* text boxes to adjust the percentages of each primary color for the basic color you selected to the RGB values you want to use for your custom color.**

Border Attributes

On the **Border** property page (Figure 4-9), you can define attributes for the borders of table cells.

<i>ATTRIBUTE</i>	<i>DESCRIPTION</i>
Color	Sets the color of the border for the cells in a table. Enter the RGB values of the color you want to use in the Red, Green, and Blue fields, or click Custom Color to select a color from the Color dialog.
Width	Sets the width of the table cells.

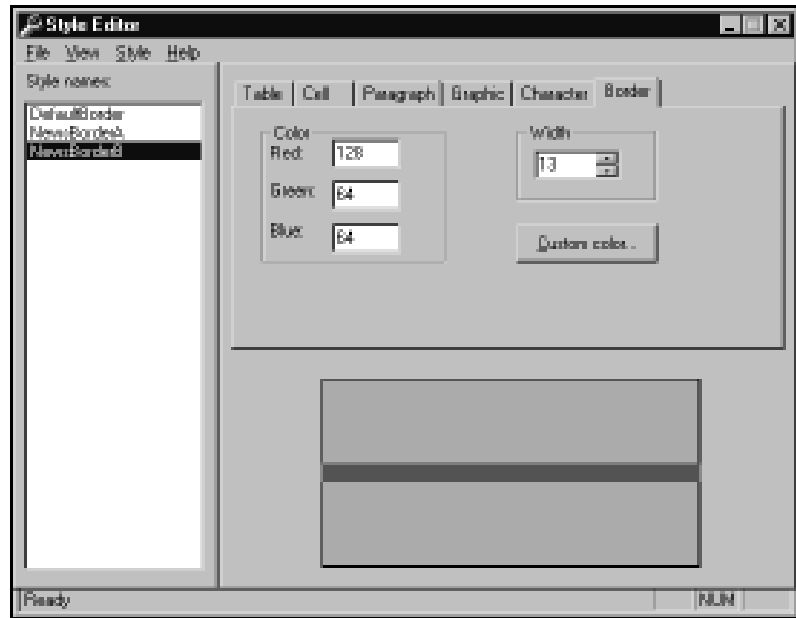


Figure 4-9
Border tab dialog

Modifying a Style

You can modify any style in a style file by opening the style file, selecting the appropriate tab dialog, and editing the cell style (or whichever style) you want to edit. You can then choose **Save** from the **File** menu to save your changes.

You can change the particular combination of styles you have assigned to **MasterStyle** by opening the **Table** tab dialog and changing the current styles selected for Cell, Paragraph, Graphic, and Border. You can also change the character style associated with any given paragraph style in the **Paragraph** tab dialog.

Editing Attributes

You can edit a style's attributes by opening the appropriate tab dialog in the Style Editor and highlight the name of the style you want to edit in the **Styles** list. You can change the appropriate attributes on the tab dialog.

You might want to print an SMF file so you can review all its defined styles. That way, if you want to know the border defined for a particular style, you don't need to open *GUIDE* Table Viewer Style Editor.

Deleting a Style

To delete a style from a style file in *GUIDE* Table Viewer Style Editor:

- 1 Open the tab dialog for the type of style you want to delete.**
- 2 Select the specific style you want to delete from the *Styles* list.**
- 3 Choose *Delete* from the *Styles* menu.**

The Style Editor prompts you to confirm the deletion. When you click on **OK**, the Style Editor deletes the style and removes its name from the **Styles** list.

- 4 Choose *Save* from the *File* menu.**

This updates the SMF file.

INDEX

Symbols

.smf extension 14
 .tmf extension 14
 <cell> tag 19, 25
 <colspec> tag 20, 21
 <graphic> tag 17, 24
 <item> tag 22
 <list> tag 17
 <para> tag 17
 <row> tag 19
 <table> tag 17, 18
 <tfoot> tag 17
 <ttile> tag 17

A

Add Style dialog 42
 adding a new style to a style file 42
 attributes
 bdrflag 26
 border 21
 bulletchar 22
 columns 18
 file 24
 for Border style 50
 for Cell style 44
 for Character style 48
 for Graphic style 46
 for Paragraph style 46
 for Table style 44
 rows 18

attributes (*continued*)
 separator 22
 splength 25
 sptype 25
 stlfile 33
 style 22, 43
 width 21

B

basic concepts about tables 14
 bdrflag attribute 26
 body of table 16
 border attribute 21
 Border style 31
 attributes of 50
 bulletchar attribute 22

C

Cell style 31, 36
 attributes of 44
 cells
 adding lists to 22
 adding paragraphs to 21
 basic concepts 15
 lists in 22
 merging to create spans 25
 character references, support for 27

Character style 31, 36
 attributes of 48
columns attribute 18
command, Place Graphic 29
conventions for documentation 10
creating a style 42

D

default styles 35
defining styles 39
defining table content 21
Document Type Definition. *See* DTD
documentation conventions 10
DTD 9
 SMF 32
 TMF 9

E

editing style attributes 52
error message when placing tables 30

F

file attribute 24
File menu 37
font
 selecting non-system 49
 size range 49
footer component in tables 16, 27
formatting with the Style Editor 31

G

Graphic style 31, 36
 attributes of 46
graphics formats in tables 24
graphics, placing in a table 24, 29
GUIDE Table Viewer Style Editor. *See* Style Editor
guideio.dll 11
guidetv.dll 9, 11

H

Help menu 39
Hotspots in tables 16

I

itemizer attribute 22

K

keyboard shortcuts in Style Editor 37

L

lists, adding to cells 22

M

MasterStyle
 default styles in 35
 explained 35
 menus
 File 37
 Help 39
 Style 39
 View 38

N

non-system font, selecting 49

P

Paragraph style 31, 36
 attributes of 46
 set by default 22
 paragraphs, adding to cells 21
 parser, support for character references 28
 Place Graphic command for placing
 tables 14, 29
 placing a table in GUIDE Author 29
 point size for fonts, range available 49

R

rows attribute 18

S

saving a new style file 41
 scripts, display format in manual 11
 SGML 9, 32
 shortcuts, keyboard 37
 size of fonts, range available 49
 SMF 9, 14
 SMF DTD 32
 SMF file
 conflict with TMF file 40
 creating 14, 31
 explained 14
 extract from 32
 spans 25
 splength attribute 25
 sptype attribute 25
 Standard Generalized Markup Language. *See*
 SGML
 stlfile attribute 33
 style
 adding to a style file 42
 creating 42
 style attribute 22
 style attributes 43
 changing 43
 defining 43
 editing 43, 52
 for Border style 50
 for Cell style 44
 for Character style 48
 for Graphic style 46
 for Paragraph style 46
 style conflicts, resolving 40

- Style Editor 9
 - defining styles in 31
 - explained 9
 - formatting with 31
 - interface 34, 36
 - keyboard shortcuts in 37
 - menus 36
 - starting 33
 - Styles List 34
 - styling elements in 9
- style file 32
 - adding a new style to 42
 - creating 31, 40
 - referencing in a TMF file 41
 - saving new 41
- Style Markup Format. *See* SMF
- Style menu 39
- styles 31, 36
 - assigning attributes to 42
 - Border 31
 - calling from the database 33
 - Cell 31, 36
 - Character 31, 36
 - defining 31, 39
 - deleting 52
 - Graphic 31, 36
 - Paragraph 31, 36
 - selecting a non-system font 49
 - Table 31, 36
- Styles list 34
- subordinate tags 17
- syntax, convention for displaying 10

T

- table
 - creating the body of 19
 - creating with a TMF file 13
 - displaying in GUIDE 13
 - largest possible in GUIDE 19
- table content
 - defining 21
 - explained 13
- Table Markup Format. *See* TMF
- table structure 16
- Table style 31, 36
 - attributes of 44
- table styles, creating 31
- Table Viewer DLL
 - explained 9
 - installation of 11
- tables
 - adding Hotspots to 13
 - basic concepts 14
 - body of 16
 - defining content 21
 - footer component 16, 27
 - graphics formats in 24
 - Hotspots in 16
 - interactive possibilities of 13
 - links to 13
 - list content 23
 - placing 29
 - placing in GUIDE Author 29
 - styling elements in 9
 - title component 16, 26
 - TMF for basic structure 17
 - treatment of listed items 23
 - types of content 16

tags

<cell> 19, 25

<colspec> 20, 21

<graphic> 17, 24

<item> 22

<list> 17, 22

<para> 17

<row> 19

<table> 17, 18

<tfoot> 17

<tttitle> 17

subordinate 17

used in TMF 17

title component 16, 26

TMF 9

for basic table structure 17

parser 27

tags used 17

using to generate tables 9

TMF file, how to create 13

TMF tags in the DTD 13

types of styles 36

V

View menu 38

W

width attribute 21

Table Viewer DLL™

TMF Reference

***Table Markup Format™ Reference
for GUIDE® Author™***



Contents

ABOUT THIS REFERENCE

About the Tag Descriptions 63

The TMF Parser 65

About Sample Scripts 66

TABLE MARKUP FORMAT TAG DESCRIPTIONS

CELL 67

COLSPEC 68

GRAPHIC 70

ITEM 71

LIST 73

PARA 75

ROW 76

SO 78

TABLE 80

TFOOT 82

TTITLE 83

APPENDIX - TABLE MARKUP FORMAT DTD 85

ABOUT THIS REFERENCE

With Table Viewer DLL you can display tabular data from a Table Markup Format_{TM} (TMF_{TM}) file in a *GUIDE* document. Any table you render derives its organization from a hierarchical structure specified in the TMF document type definition (DTD). A printed version of the DTD is included as an appendix to this reference; an electronic version can be found in the DTD folder in your *GUIDE Author* installation directory. This reference elaborates on the TMF tags listed in the DTD.

A TMF file is simply an ASCII file that defines table structure and organization by use of the tags and syntax outlined in this section and explained further in the TMF Tag Descriptions section that follows. The alphabetical list of tags defines each tag, explains its allowable subordinate tags and any applicable attributes, and provides an example of the tag's use.

This reference assumes you are familiar with DTD structure and Standard Generalized Markup Language. TMF files typically contain references to SMF files, which determine how a table will actually display on-line. For more information on SMF files, see the *Style Markup Format Reference*, which follows this reference.

About the Tag Descriptions

The description of each style tag in this reference is organized under four main headings: Syntax, Tag Definition, Subordinate Tags, and Attributes. Under a fifth heading, Example, can be found sample syntax for the tag described. For ease of review, tags are set in upper-case bold except for the sample syntax under Example. There, they are set in lower-case. Since tags are not case-sensitive, it doesn't matter. It also doesn't matter whether tag attributes are placed between single or double quotation marks. In the sample syntax we use double quotes.

As you review the individual tags in this reference, keep in mind how they can be grouped for ease of understanding. The **TABLE**, **COLSPEC**, **ROW**, and **CELL** tags define a table's basic structure. The **TTITLE** and **TFOOT** tags are optional, but if used would create title and footer components for the table. Once the basic structure of the table is set up with these tags, you then populate the individual cells in the table with paragraphs of text (the **PARA** tag), lists (the **LIST** and **ITEM** tags), and/or graphics (the **GRAPHIC** tag).

Syntax

The syntax section is a quick reference to the table markup language. In the following example, the first word inside the < character must be the table element being defined. All words that follow up to the > character are the allowable attributes of the element. The attribute terms and their values are placed within brackets ([]). Since most attributes are optional, you can generally use as few (including none) or as many of these attributes as you wish. (The Tag Descriptions section notes those attributes that are required.)

For ease of reference, attributes appear in all capital letters, for example, **STYLE**. If [Subordinate Tags] is placed before an element's closing tag, refer to the element's Subordinate Tags section to see which ones are allowable. For example, the **TTITLE** subordinate tag becomes a subordinate tag of the **TABLE** tag when you create a title component for a table. Element tag syntax concludes with the end tag markup (</...>), where you repeat the element name after the slash, for example, </table>.

```
<table columns="3" rows="16" stlfile="allstyle.smf"
bdrleft="Border2" bdrright="Border2" Bdrtop="Border2"
bdrbtm="Border2">
  <colspec column="1" border="Border2">
  <colspec column="2" border="Border2">
  <colspec column="3" border="Border2">
  [Subordinate tags]
</table>
```

Definition

The definition explains the purpose of the element; for example, the **TTITLE** tag corresponds to the title component of a table.

Subordinate Tags

A list of subordinate tags identifies other tags that can be placed inside the tag currently being defined. For example, the **ROW** tag can include a **CELL** tag but not a **COLSPEC** tag, while the **CELL** tag can include the **PARA**, **LIST**, and **GRAPHIC** tags that represent the three types of content you can place in a table.

Attributes

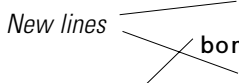
Attribute values define each element in more detail. Some attributes are required, while others are optional (if necessary, Table Viewer DLL will provide a default value if you do not assign one). Measurement values are in TWIPs (twentieths of a point; 1440 TWIPs equal one inch).

The TMF Parser

The TMF parser can interpret character references, which use numbers based on ASCII or ANSI codes to represent keyboard characters. The code numbers must be preceded by the ampersand (&) and pound (#) symbols. For example, typing **&#60** instructs the parser to insert a < character into the content.

About Sample Scripts

A word about sample scripts shown in this reference. Where printed lines end on these pages isn't necessarily where script lines break on the computer screen. You'll recognize continuation lines that belong with the preceding lines because the continuation lines return to the left margin, not to the indentation level where the script line began. The first level of indentation shown in these pages represents the left margin of a computer screen. For clarity, even though scripts aren't case-sensitive, we've used upper- and lower-case characters in the scripts shown in this manual for better readability. Here is an example of what we mean.

New lines  `<cell rownum="100" height="1500" border="Border2">
A cell in row 100 with a height of 1500 TWIPs and a
border set with the Border2 style.
</cell>`

Continuation line: On the screen, these lines would be part of the previous line; note that they return to the left margin, not the indent level where the script started.

TABLE MARKUP FORMAT TAG DESCRIPTIONS

Following, in alphabetical order, is a description of each TMF tag. Note, in particular, the **STYLE** attribute, which can be used to override default styles set by the style file.

CELL

Syntax

```
<CELL [STYLE="Named Style"] [COLUMN="Number"] [SPTYPE="horiz | vert"]
[SPLENGTH="Number"]>
  [Subordinate tags]
</CELL>
```

Tag Definition

The **CELL** tag identifies a cell at a specific row and column; it may also be the first cell in a horizontal or vertical span of cells.

Subordinate Tags

The **CELL** tag can contain any of the following subordinate tags:

PARA

LIST

GRAPHIC

*CELL (continued)**Attributes*

<i>NAME</i>	<i>TYPE</i>	<i>USE</i>	<i>DESCRIPTION</i>
STYLE	Character string	Optional	Specifies a style for the cell that would override the default cell style specified for the table
COLUMN	Number	Optional	Specifies the number of the column to which the cell belongs; if the number is not stated, it is implicit in the number of cells defined before this one inside a ROW tag
SPTYPE	horiz vert	Optional	Specifies whether a span, if any, is horizontal or vertical
SLENGTH	Number	Optional	Specifies the number of cells that comprise the span

Example

```
<cell sptype="horiz" splength="3">
</cell>
```

COLSPEC*Syntax*

```
<COLSPEC [COLUMN="Number"] [WIDTH="Number"] [BORDER="Character
string"]>
```

TMF Tag Descriptions

COLSPEC (continued)

Tag Definition

The **COLSPEC** tag marks a column defined within the table. It is used to provide attributes to a specified column. It can force a column to have a particular width or define the style of the right border of that column. The **COLSPEC** tag has no closing tag.

Subordinate Tags

The **COLSPEC** tag cannot contain any subordinate tags.

Attributes

NAME	TYPE	USE	DESCRIPTION
COLUMN	Number	Optional	Identifies the specific column being defined
WIDTH	Number	Optional	Specifies the width of the column in TWIPs; overrides the row width that is usually set by the widest cell in the column as determined by cell content
BORDER	Character string	Optional	Specifies the style of the right border

Example

```
<colspec column="1" width="50" border="Border2">
```

GRAPHIC

Syntax

```
<GRAPHIC [STYLE="Named Style"] [FILE="Character string"]>
</GRAPHIC>
```

Tag Definition

The **GRAPHIC** tag marks a graphic element. The closing tag (**</graphic>**) is optional.

Subordinate Tags

The **GRAPHIC** tag cannot contain any subordinate tags.

Attributes

NAME	TYPE	USE	DESCRIPTION
STYLE	Character string	Optional	Specifies a style that defines the alignment and indent of the graphic and would override the default graphic style.
FILE	Character string	Required	Specifies the external reference for the graphic file to be placed in the cell

GRAPHIC (continued)

Example

```

<row>
<cell>
<graphic style="graph" file="c:\pics\switch.bmp">
</graphic>
</cell>
</row>

```

ITEM

Syntax

```

<ITEM>
[Subordinate tags]
</ITEM>

```

Tag Definition

The **ITEM** tag marks a specific entry for a list. The entry can be a paragraph or graphic, as defined by the appropriate tag. Since you can nest lists, the **ITEM** tag may also signify the start of another list.

ITEM (continued)

Subordinate Tags

The **ITEM** tag can contain any of the following subordinate tags:

PARA

LIST

GRAPHIC

Attributes

None.

Example

```
<list>
<item>
<para>
List Item 1
</para>
</item>
<item>
<para>
List Item 2
</para>
</item>
</list>
```

LIST

Syntax

```
<LIST [STYLE="Named Style"] [ITEMIZER="Numeric | UpperAlpha | LowerAlpha | UpperRoman | LowerRoman | Bullet"] [ITWIDTH="Number"] [SEPARATOR="Character string"] [BULLETCCHAR="Character string"]>
  [Subordinate tags]
</LIST>
```

Tag Definition

The **LIST** tag marks a list element.

Subordinate Tags

The **LIST** tag must contain the subordinate tag **ITEM** in order to define list content.

Attributes

NAME	TYPE	USE	DESCRIPTION
ITEMIZER	Numeric Bullet UpperAlpha LowerAlpha UpperRoman LowerRoman	Optional	Specifies the type of number or bullet for the list

LIST Attributes (continued)

NAME	TYPE	USE	DESCRIPTION
ITWIDTH	Number	Optional	Defines the width of the area the itemizer occupies, in TWIPs.
SEPARATOR	Character string	Optional	Specifies the separator to be placed between the list item number or bullet and the item content
BULLETCCHAR	Character string	Optional	Specifies the style of the bullet to be used with the Bullet itemizer

The **ITEMIZER** attribute produces a numbered or bulleted list as follows:

Numeric	1, 2, 3,
Bullet	·, ·, ·,
UpperAlpha	A, B, C,
LowerAlpha	a, b, c,
UpperRoman	I, II, III,
LowerRoman	i, ii, iii

Example

```

<list itemizer="UpperAlpha" itwidth="50"
separator=".">
  <item>
    <para>
      List Item #1
    </para>
  </item>

  <item>
    <para>
      List Item 2
    </para>
  </item>

  <item>
    ...
  </item>
</list>

```

PARA

Syntax

```
<PARA> [STYLE="Named Style"]  
</PARA>
```

Tag Definition

The **PARA** tag marks the actual text being placed in a cell.

Subordinate Tags

The **PARA** tag cannot contain any subordinate tags.

Attributes

NAME	TYPE	USE	DESCRIPTION
STYLE	Character string	Optional	The named style overrides the default paragraph style and defines the appearance of the paragraph in terms of the font (name, size, bold, italic, etc.)

PARA (continued)

Example

```
<cell>
<para style="Font2">
Add a sales charge of 0.33% to purchases under $25,000
</para>
</cell>
```

ROW

Syntax

```
<ROW [STYLE="Named Style"] [ROWNUM="Number"] [HEIGHT="Number"]
[BORDER="2"]>
  [Subordinate tag]
</ROW>
```

Tag Definition

The **ROW** tag marks a new row in a table.

*ROW (continued)**Subordinate Tags*

The **ROW** tag can contain the **CELL** subordinate tag.

Attributes

NAME	TYPE	USE	DESCRIPTION
STYLE	Character string	Optional	Specifies a style that defines the row and overrides the default row style
ROWNUM	Number	Optional	Identifies a unique cell by its number in the row; if row numbers are not stated, the numbering implicitly starts with 1 and increases as new rows are added.
HEIGHT	Number	Optional	Specifies the row height in TWIPs; overrides the row height that is usually set by the tallest cell in the row as determined by cell content
BORDER	Character string	Optional	Specifies the style of the row border

Example

```
<row>
<cell rownum="10" height="1500" border="Border2">
Throttle position sensor (TPS)
</cell>
</row>
```

SO

Syntax

```
<SO [FONTNAME="Character string"] [SIZE="Number"]  
[BOLD="y1 | n1"] [ITALIC="y2 | n2"] [UNDERL="y3 | n3"]  
[STRIKE="y4 | n4"] [SUPER="y5 | n5"] [SUB="y6 | n6"]  
[SOCOLOR="HEX value"]>  
  </SO>
```

Tag Definition

The **SO** tag overrides style attributes of the current paragraph style. When the **SO** closing tag is encountered, the paragraph reverts to the named or default paragraph style.

Subordinate Tags

The **SO** tag cannot contain any subordinate tags.

Attributes

NAME	TYPE	USE	DESCRIPTION
FONTNAME	Character string	Optional	Specifies the name of the font to be used in the style override.
SIZE	Number	Optional	Specifies the font size

TMF Tag Descriptions

SO Attributes (continued)

NAME	TYPE	USE	DESCRIPTION
BOLD	y1 n1	Optional	Specifies the use of bold: yes=y1; no=n1 (the default)
ITALIC	y2 n2	Optional	Specifies the use of italic: yes=y2; no=n2 (the default)
UNDERL	y3 n3	Optional	Specifies the use of underline: yes=y3; no=n3 (the default)
STRIKE	y4 n4	Optional	Specifies the use of strikethrough: yes=y4; no=n4 (the default)
SUPER	y5 n5	Optional	Specifies the use of superscript: yes=y5; no=n5 (the default)
SUB	y6 n6	Optional	Specifies the use of subscript: yes=y6; no=n6 (the default)
SOCOLOR	Number	Optional	Specifies the foreground text color using a HEX value; examples: "0xFF0000" is blue, "0x00FF00" is green, and "0x0000FF" is red.

Example

```

<para>
  This door <so fontname="Arial" Size="16"
  socolor="0x0000FF">must</so>be closed and locked when you
  leave the laboratory.
</para>

```

Note how “must” is surrounded by the opening and closing **SO** tags. As a result, “This door” would be in the default paragraph style; “must” would be set in Arial, 16 point, and colored red; and “be closed and locked when you leave the laboratory” would revert to the default paragraph style.

TABLE

Syntax

```
<TABLE [COLUMNS="Number"] [ROWS="Number"] [STLFILE="Named style"]  
[BDRLEFT="Character string"] [BDRTOP="Character string"]  
[BDRRIGHT="Character string"] [BDRTM="Character string"]>  
  [Subordinate tags]  
</TABLE>
```

Tag Definition

The **TABLE** tag marks a table element and cannot be repeated.

Subordinate Tags

Any combination of the following subordinate tags may be used (the **TTITLE** and **TFOOT** tags only once)

COLSPEC

TTITLE

ROW

TFOOT

*TABLE (continued)**Attributes*

<i>NAME</i>	<i>TYPE</i>	<i>USE</i>	<i>DESCRIPTION</i>
COLUMNS	Number	Optional	Specifies the number of columns in the table
ROWS	Number	Optional	Specifies the number of rows in the table
STLFILE	Character string	Optional	Specifies the style (SMF) file to be applied to the table
BDRLEFT	Character string	Optional	Specifies the style of the left border of the table
BDRTOP	Character string	Optional	Specifies the style of the top border of the table
BDRRIGHT	Character string	Optional	Specifies the style of the right border of the table
BDRBTM	Character string	Optional	Specifies the style of the bottom border of the table

Example

```

    <table columns="10" rows="16" stlfile="allstyle.smf"
    bdrleft="Border2" bdrright="Border2" bdrtop="Border2"
    bdrbtm="Border2">
    <tttitle style="TitleStyle">
    <para>
    Policies & Procedures
    </para>
    </tttitle>
    ...
    </table>

```

TFOOT

Syntax

```
<TFOOT [STYLE="Named Style"] [HEIGHT="Number"] [BDRFLAG="Y1 | N1"]>  
[Subordinate tag]  
</TFOOT>
```

Tag Definition

The **TFOOT** tag marks a table footer element.

Subordinate Tags

The **TFOOT** tag can contain the following subordinate tags:

PARA **LIST** **GRAPHIC**

Attributes

<i>NAME</i>	<i>TYPE</i>	<i>USE</i>	<i>DESCRIPTION</i>
STYLE	Character string	Optional	Specifies a style that overrides the default style for the table
HEIGHT	Number	Optional	Specifies the height of the footnote component in TWIPs.

TMF Tag Descriptions

TFOOT Attributes (continued)

NAME	TYPE	USE	DESCRIPTION
BDRFLAG	y1 n1	Optional	Specifies whether the footer has a border: yes=y1 (the default); no=n1; if there is a border, it takes on the style of the left, bottom, and right borders defined in the TABLE tag

Example

```
<tfoot height="1440" brdflag="n1">
</tfoot>
```

TTITLE

Syntax

```
<TTITLE [STYLE="Named Style"] [HEIGHT="Number"] [BDRFLAG="y1 | n1"]>
[Subordinate tag]
</TTITLE>
```

Tag Definition

The **TTITLE** tag marks a table title.

*TTITLE (continued)**Subordinate Tags*

The **TTITLE** tag can contain the following subordinate tags:

PARA **LIST** **GRAPHIC**

Attributes

<i>NAME</i>	<i>TYPE</i>	<i>USE</i>	<i>DESCRIPTION</i>
STYLE	Character string	Optional	Specifies a style that overrides the default style for the table
HEIGHT	Number	Optional	Specifies the height of the title component in TWIPs.
BDRFLAG	y1 n1	Optional	Specifies whether the title component has a border: yes=y1 (the default); no=n1. If there is a border, it takes on the style of the left, top, and right borders as defined in the TABLE tag.

Example

```
<ttitle height="960" bdrflag="n1">  
<para>  
Power Supply  
</para>  
</ttitle>
```

APPENDIX

TABLE MARKUP FORMAT DTD

```

<! Tables TMF (Table Markup Format) DTD  >
<!-- Entity declarations -->
<!entity % tabcont      " ((ttitle)?, (colspec)*, (row)*, (tfoot)? ) " >
<!entity % content      " ( para | list | graphic)* " >
<!entity % cattrib      " style      CDATA #IMPLIED"      >

<! Table structure  >

<! Table element  >
<!element table --      ( %tabcont; ) >
<!attlist table
      columns      NUMBER      #IMPLIED
      rows         NUMBER      #IMPLIED
      stlfile      CDATA       #IMPLIED
      bdrleft      CDATA       #IMPLIED
      bdrtop       CDATA       #IMPLIED
      bdrright     CDATA       #IMPLIED
      bdrbtm       CDATA       #IMPLIED>

<! Title element  >
<!element ttitle --      ( %content; ) >
<!attlist ttitle
      style      CDATA      #IMPLIED
      height     NUMBER     #IMPLIED
      bdrflag    (y1 | n1)  "y1">

<! Column defined within the scope of a table  >
<!element colspec -- o EMPTY >
<!attlist colspec
      column      NUMBER      #IMPLIED
      width       NUMBER      #IMPLIED
      border      CDATA       #IMPLIED>

<!-- specifier for a new row in a table -->
<!element row --      ( (cell)* ) >
<!attlist row
      style      CDATA      #IMPLIED
      rownum     NUMBER     #IMPLIED
      height     NUMBER     #IMPLIED
      border     CDATA      #IMPLIED>

```

```

<! Footer element >
<!element tfoot - - ( %content; ) >
<!attlist tfoot
    style          CDATA          #IMPLIED
    height         NUMBER         #IMPLIED
    bdrflag        (y1 | n1)      "y1">

<!-- Cell defined within the scope of a row or a column -->
<!element cell - - ( %content; ) >
<!attlist cell
    style          CDATA          #IMPLIED
    column         NUMBER         #IMPLIED
    sptype         (horiz | vert)  #IMPLIED
    splength       NUMBER         #IMPLIED>

<!-- Paragraph element -->
<!element para - - ( (#PCDATA | so)* ) >
<!attlist para
    style          CDATA          #IMPLIED

<!-- Style override element -->
<!element so - - ( %content; )>
<!attlist so
    fontname       CDATA          #IMPLIED
    size           NUMBER         #IMPLIED
    bold           ( y1 | n1 )    "n1"
    italic         ( y2 | n2 )    "n2"
    underl        ( y3 | n3 )    "n3"
    strike         ( y4 | n4 )    "n4"
    super         ( y5 | n5 )    "n5"
    sub           ( y6 | n6 )    "n6"
    socolor        CDATA          #IMPLIED>

<! Graphic element >
<!element graphic - o EMPTY >
<!attlist graphic
    style          CDATA          #IMPLIED
    file           CDATA          #REQUIRED>

<! List element >
<!element list - - ( (item)* ) >
<!attlist list
    Itemizer       (Numeric | Bullet | #IMPLIED
                   UpperAlpha | LowerAlpha |
                   UpperRoman | LowerRoman)
    Itwidth        Number         #IMPLIED
    Separator      CDATA          #IMPLIED
    BulletChar     CDATA          #IMPLIED>

<! List-item element >
<!element item - - ( %content; ) >
<!--End of Elements -->

```

Table Viewer DLL™

***Style Markup Format™ Reference
for GUIDE® Author™***

SMF Reference

Contents

ABOUT THIS REFERENCE

About the Style Tag Descriptions	91
Tag Definition	92
Subordinate Tags	92
Attributes	92
Examples	92
About Sample Scripts	93

STYLE MARKUP FORMAT TAG DESCRIPTIONS

BORDER	95
CELLSTL	96
CHARSTL	97
COLOR	98
GRASTL	99
PARASTL	100
STLCOL	102
TABLE	103

APPENDIX - STYLE MARKUP FORMAT DTD	105
--	-----

ABOUT THIS REFERENCE

With *GUIDE* Table Viewer Style Editor, you can determine how a table is rendered on-screen by creating style files that make use of Style Markup Format™ (SMF™). Style or SMF files, which are referenced from the TMF files you use to place tables in *GUIDE* documents, are created from the tags found in the SMF document type definition (DTD). A printed version of the DTD is included as an appendix to this reference; an electronic version can be found in your *GUIDE Author* installation directory. This reference elaborates on the SMF tags listed in the DTD.

Like TMF files, SMF files are ASCII files. But whereas TMF files determine table structure and organization, SMF files control the on-screen appearance of tables. This is done with the tags outlined in this section and explained further in the SMF Tag Descriptions section that follows. The alphabetical list of tags defines each tag, explains its related subordinate tags and applicable attributes, and provides an example of the tag's use.

This reference assumes you are familiar with DTD structure and Standard Generalized Markup Language. However, since *GUIDE* Table Viewer Style Editor reads and writes SMF files, it is not essential to understand their exact syntax. For more information on *GUIDE* Table Viewer Style Editor, see Chapter 4 of the *Table Viewer DLL User's Manual*. For more information about TMF files, see the *Table Markup Format Reference* that precedes this reference.

About the Style Tag Descriptions

Different styles determine the appearance of tables when they are rendered in a *GUIDE* window. The styles that can be set in an SMF file are for table, cell, paragraph, graphic, character, and border. The "table style" isn't actually a style at all, but a collection of the five other styles. That is, when you create a table style, you actually create a **MasterStyle** for a style file consisting of a single style drawn from each of the other five style types.

The description of each style tag in this reference is organized under three main headings: Tag Definition, Subordinate Tags, and Attributes. Under a fourth heading, Example, can be found sample syntax for the tag described. For ease of review, tags are set in upper-case bold under Tag Definition, Subordinate Tags, and Attributes. In the sample syntax under Example, however, they are set in lower-case, which is how *GUIDE* Table Viewer Style Editor treats tags when it generates the style files you create. Since tags are not case-sensitive, it doesn't matter. It also doesn't matter whether tag attributes are placed between single or double quotation marks. In the sample syntax we use double quotes; the Style Editor generates single quotes.

Tag Definition

The tag definition explains the purpose of the element being described; for example, the **CHARSTL** tag marks the beginning of the parameter and attribute values that define a character style.

Subordinate Tags

Subordinate tags are parameters that can be placed inside the beginning and ending tags of the table element being defined. For example, the **CELLSTL** tag used to define a cell style could include the **COLOR** tag.

Attributes

Attributes define a table element in more detail. Attribute information includes allowable values such as whether the value is required or optional. If required, this reference states the attribute's default value.

Examples

These provide sample syntax of the tag in use. See the next section.

About Sample Scripts

A word about sample scripts shown in this reference. Where printed lines end on these pages isn't necessarily where script lines break on the computer screen. You'll recognize continuation lines that belong with the preceding lines because the continuation lines return to the left margin, not to the indentation level where the script line began. The first level of indentation shown in these pages represents the left margin of a computer screen. For clarity, even though scripts aren't case-sensitive, we've used upper- and lower-case characters in the scripts shown in this manual for better readability. Here is an example of what we mean.

New lines 

```
<cell rownum="100" height="1500" border="Border2">  
A cell in row 100 with a height of 1500 TWIPs and a  
border set with the Border2 style.  
</cell>
```

Continuation line: On the screen, these lines would be part of the previous line; note that they return to the left margin, not the indent level where the script started.

STYLE MARKUP FORMAT TAG DESCRIPTIONS

BORDER

The **BORDER** tag marks the border style for a cell.

Subordinate Tags

The **BORDER** tag can contain the **COLOR** subordinate tag.

Attributes

NAME	TYPE	USE	DESCRIPTION
ID	Character string	Required	Specifies a unique identifier for the border style
WIDTH	Number	Optional	Specifies the width of the border, in pixels; default=1

Example

```
<borderstl id="DefaultBorder" width="1"
</borderstl>
```

CELLSTL

Tag Definition

The **CELLSTL** tag marks a cell style.

Subordinate Tags

The **CELLSTL** tag can contain the **COLOR** subordinate tag.

Attributes

NAME	TYPE	USE	DESCRIPTION
ID	Character string	Required	Specifies a unique identifier for the cell style

Example

```
<cellstl id=CellStyle>  
<color red="64" green="0" blue="0">  
</color>  
</cellstl>
```

CHARSTL

Tag Definition

The **CHARSTL** tag marks a character style.

Subordinate Tags

The **CHARSTL** tag can contain the **COLOR** subordinate tag.

Attributes

NAME	TYPE	USE	DESCRIPTION
ID	Character string	Required	Specifies a unique identifier for the character style
FONTNAME	Character string	Required	Specifies the name of the font for the style
SIZE	Character string	Required	Specifies the point size for the font
BOLD	y1 n1	Optional	Specifies bold: yes=y1; no=n1 the default)
ITALIC	y2 n2	Optional	Specifies italic: yes=y2; no=n2 (the default)
UNDERL	y3 n3	Optional	Specifies underline: yes=y3; no=n3 (the default)
STRIKE	y4 n4	Optional	Specifies strikethrough: yes=y4; no=n4 (the default)

CHARSTL Attributes (continued)

NAME	TYPE	USE	DESCRIPTION
SUPER	y5 n5	Optional	Specifies surperscript: yes=y5; no=n5 (the default)
SUB	y6 n6	Optional	Specifies subscript: yes=y6; no=y6 (the default)

Example

```
<charstl id="CharA" fontname="Arial" size="10"
bold="y1" italic="n2" underl="y3" strike="n4" super="n5"
sub="n6">
</charstl>
```

COLOR*Tag Definition*

The **COLOR** tag marks a color for the tag it is contained in.

Subordinate Tags

The **COLOR** tag cannot contain any subordinate tags.

SMF Tag Descriptions

COLOR (continued)

Attributes

NAME	TYPE	USE	DESCRIPTION
RED	Number	Optional	Specifies an RGB value for red
GREEN	Number	Optional	Specifies an RGB value for green
BLUE	Number	Optional	Specifies an RGB value for blue

Example

```
<color red="0" green="0" blue="240">
</color>
```

GRASTL

Tag Definition

The **GRASTL** tag marks a graphic style.

Subordinate Tags

The **GRASTL** tag cannot contain any subordinate tags.

GRASTL (continued)

Attributes

<i>NAME</i>	<i>TYPE</i>	<i>USE</i>	<i>DESCRIPTION</i>
ID	Character string	Required	Specifies a unique identifier for the graphic style
ALIGN	Character string	Optional	Specifies the alignment of the graphic in a cell: Left (indented from the left), Center, or Right (indented from the right); the default is Left.
INDENT	Number	Optional	Specifies the indentation of the graphic within the cell, depending on the alignment selection

Example

```
<grastl id="DefaultGraphic" align="left" indent="0">
</grastl>
```

PARASTL

Tag Definition

The **PARASTL** tag marks a paragraph style.

PARASTL (continued)

Subordinate Tags

The **PARASTL** tag cannot contain any subordinate tags.

Attributes

NAME	TYPE	USE	DESCRIPTION
ID	Character string	Required	Specifies a unique identifier for the paragraph style
CHARID	Character string	Required	Specifies a unique identifier for the character style within the paragraph style
ALIGN	Character string	Optional	Specifies the alignment of text in cells: Left (the default), Center, or Right
FIRSTIND	Number	Optional	Specifies the indentation of the first line of a paragraph
INDENT	Number	Optional	Specifies the indentation of all lines in a paragraph (from the left)

Example

```
<parastl id="DefaultPara" charid="DefaultChar"
align="left" firstind="0" indent="0">
</parastl>
```

STLCOL

Tag Definition

The **STLCOL** tag marks a collection of styles. Every SMF file opens and closes with this tag.

Subordinate Tags

The **STLCOL** tag can contain the following subordinate tags:

TABLESTL	CELLSTL	PARASTL
GRASTL	CHARSTL	BORDERSTL

Attributes

The **STLCOL** tag has no attributes.

Example

```
<stlcol>
  <tablestl id="MasterStyle" cell="CellStyle"
para="ParaStyle" graphic="GraphicStyle"
border="BorderStyle">
  </tablestl>

  ....
</stlcol>
```

TABLE

Tag Definition

The **TABLE** tag marks a table style.

Subordinate Tags

The **TABLE** tag cannot contain any subordinate tags.

Attributes

NAME	TYPE	USE	DESCRIPTION
ID	Character string	Required	Specifies a unique identifier for a table style
CELL	Character string	Required	Specifies a default cell style
PARA	Character string	Required	Specifies a default paragraph style (which includes a default character style)
GRAPHIC	Character string	Required	Specifies a default graphic style
BORDER	Character string	Required	Specifies a default border style

Example

```
<tablestl id="MasterStyle" cell="DefaultCell"
para="ParaA" graphic="Graph6" border="DefaultBorder">
</tablestl>
```


APPENDIX

STYLE MARKUP FORMAT DTD

```
<!-- Tables SMF (Style Markup Format) DTD -->
<!-- Collection of styles -- >

<!element stlcol -- ( (tablestl)      & (cellstl)+      &
                    (parastl)+        & (grastl)+        &
                    (charstl)+        & (bordrstl)+      )>

<!-- Table style element -->
<!element tablestl - o EMPTY>
<!attlist
    id          CDATA          #REQUIRED
    cell        CDATA          #REQUIRED
    para        CDATA          #REQUIRED
    graphic     CDATA          #REQUIRED
    border      CDATA          #REQUIRED>

<!-- Cell style element -->
<!element cellstl -- (color)>
<!attlist
    id          CDATA          #REQUIRED>

<!-- Paragraph style element -->
<!element parastl - o EMPTY>
<!attlist
    id          CDATA          #REQUIRED
    charid      CDATA          #REQUIRED
    align       ( left | center | right ) "left"
    firstind    NUMBER         #IMPLIED
    indent      NUMBER         #IMPLIED>

<!-- Graphic style element -->
<!element grastl - o EMPTY>
<!attlist
    id          CDATA          #REQUIRED
    align       ( left | center | right ) "left"
    indent      NUMBER         #IMPLIED>
```

```

<!-- Character style element -->
<!--element charstl -- (color)>
<!--attlist
            id          CDATA          #REQUIRED
            fontname    CDATA          #REQUIRED
            size        NUMBER        #REQUIRED
            bold        ( y1 | n1 )    "n1"
            italic      ( y2 | n2 )    "n2"
            underl      ( y3 | n3 )    "n3"
            strike      ( y4 | n4 )    "n4"
            super       ( y5 | n5 )    "n5"
            sub         ( y6 | n6 )    "n6">

<!-- Border style element -->
<!--element bordrstl -- (color) >
<!--attlist
            id          CDATA          #REQUIRED
            width       NUMBER        "1" >

<!-- Color style element -->
<!--element color - o  EMPTY>
<!--attlist color
            red         NUMBER        #IMPLIED
            green       NUMBER        #IMPLIED
            blue        NUMBER        #IMPLIED>

<!--END -->

```